PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 50, NO. 3–4, PP. 269–279 (2006)

# ANYTIME FOURIER ANALYSIS

Annamária R. VÁRKONYI-KÓCZY

[1] Dept. of Measurement and Information Systems,
Budapest University of Technology and Economics, Hungary
[2] Integrated Intelligent Systems Japanese-Hungarian Joint Laboratory
e-mail:koczy@mit.bme.hu

Received: January 23, 2007

## Abstract

Anytime signal processing algorithms are to improve the overall performance of larger scale embedded digital signal processing (DSP) systems. In such systems there are cases where due to abrupt changes within the environment and/or the processing system temporal shortage of computational power and/or loss of some data may occur. It is an obvious requirement that even in such situations the actual processing should be continued to insure appropriate performance. This means that signal processing of somewhat simpler complexity should provide outputs of acceptable quality to continue the operation of the complete embedded system. The accuracy of the processing will be temporarily lower but possibly still enough to produce data for qualitative evaluations and supporting decisions.

In this paper a new anytime Fourier transformation algorithm is introduced. The presented method reduces the delay problem caused by the block-oriented fast algorithms and at the same time keeps the computational complexity on relatively low level. It also makes possible the availability of partial results or estimates in case of abrupt reaction need, long or possibly infinite input data sequences.

*Keywords:* transformed domain signal processing, DFT, FFT, anytime systems.

## 1. Introduction

Computer-based monitoring and diagnostic systems are designed to handle abrupt changes due to failures within the supervised system or in its environment. The monitoring and diagnostics should be performed under prescribed response time conditions. Since these systems always work by using limited resources, it is unavoidable, even in case of extremely careful design to get into situations where the shortage of necessary data, computing power and/or processing time becomes serious, what may result in critical breakdowns.

Recently, new methods of intelligent computing have been introduced what may offer solution for avoiding these breakdowns and for keeping on processing. One of the most promising possibilities is the application of anytime techniques. The idea of anytime processing is that if there is a temporal shortage of computational power and/or there is a loss of some data the actual evaluation should be continued to provide appropriate overall performance. The solution should be processing of simpler complexity producing outputs of acceptable quality to continue the operation of the complete system. The accuracy of the processing will

be temporarily lower but possibly still enough to produce data for qualitative evaluations and supporting unavoidable decisions. Consequently, anytime algorithms provide short response time and are very flexible with respect to the available input information and computational power [1]. It is important to observe at this point that such flexibility is possible only if a vector of "shortage indicators" is an additional input of the processing facilities in use. Obviously the shortage indicators are outputs of such information processing units which monitor the actual rate of sensory data and the rate of the computational load.

Very similar problems (shortage of necessary data and/or processing time) may occur when signal processing tasks have to be performed in larger scale embedded digital signal processing (DSP) systems [2] real-time, during critical response time conditions or when the signal processing is related to feedback loops. In this latter case, concerning classical signal processing methods, block-oriented techniques have an exceptional role due to the availability of fast algorithms. However, if larger data segments are to be evaluated, the delay caused by the block-oriented approach (i.e. that we have to wait for the arrival of a complete data block before starting to process it) is not always tolerable. Thus, new approaches are needed to overcome the problem of limited (short) response time conditions. The idea of anytime techniques, proved to be useful in sharp time requirements can advantageously be used in real-time signal processing, as well.

In this paper (low complexity) block-oriented signal processing methods are combined with recursive ones. This combination flexibly reduces the delay problem caused by the block-oriented fast algorithms and at the same time keeps the computational complexity on relatively low level, i.e. offers a possibility to find a trade-off between the complexity and the delay. Furthermore, the introduced method results in an anytime behaviour of the algorithms. The early availability of partial results or estimates (even in case of possibly infinite input data sequences) opens a way to start the further processing of the data much earlier than in case of block-oriented evaluation by which the not always tolerable side-effects of processing delay can be reduced.

The paper is organized as follows: In Section II the concepts of anytime computing are briefly summarized. Section III. details the basics of block-recursive averagers. Through simple examples, Section IV. illustrates how the block-recursive averagers can be used in anytime Fourier transformation. Section V. addresses accuracy issues, while Section VI. is devoted to the conclusions.

## 2. Anytime Systems

Today there is an increasing number of applications where the computing must be carried out on-line, with a guaranteed response time and limited resources. Moreover, the available time and resources are not only limited but can also change during the operation of the system.

Good examples are the modern computer-based signal processing, diagnos-

tics, monitoring, and control systems, which are able to continuously process information, supervise complex industrial processes, and determine appropriate actions in case of failures or deviation from the optimal operational mode. In these systems the model of the supervised system is used and the evaluation of the system model must be carried out on-line. Moreover, if some abnormality occurs in the system's behaviour it may cause the reallocation of a part of the finite resources from the evaluation of the system model to another task. Also in case of an alarm signal, lower response time may be needed. Having fast approximate results can help in making early decisions for the further processing.

In these cases, the so-called anytime algorithms and systems [3] can be used advantageously, which are able to provide guaranteed response time and are flexible in respect to the available input data, time, and computational power. This flexibility makes these systems able to work in changing circumstances without critical breakdowns in the performance. Naturally, while the information processing can be maintained, the complexity must be reduced, thus the results of the computing become less accurate [4]. The models/algorithms/computing methods, which are suitable for anytime usage have to fulfil the requirements of low complexity, changeable, guaranteed response time/computational need, accuracy, and known error.

Recursive or iterative algorithms are popular tools in anytime systems, because their complexity can easily and flexibly be changed. These algorithms always give some, possibly not accurate result and more and more accurate output can be obtained if the calculations are continued. A further advantageous aspect of recursive algorithms is that we do not have to know the time/resource-need of a certain configuration in advance, the calculations can be continued until the results are needed. Then by simply stopping the evaluation, always in the given conditions achievable most accurate results are obtained. If we do not find any adequate iterative solution to a problem, other types of computing methods/algorithms can be applied in a more general modular architecture frame (for more details see [5]).

Besides the possibility of coping with the temporarily available resources and data anytime processing offers a way for supporting 'early' decision-making concerning the necessary actions for preparing the further processing or if e.g. faults/problems occur in the supervised system. With this not only the expenses of the maintenance can be reduced but in some cases also serious alarm situations can be prevented.

## 3. Block-Recursive Averagers

In this section the standard algorithms for recursive averaging are extended for data-blocks as single elements.

To illustrate the key steps first the block-recursive linear averaging will be introduced. For an input sequence $x(n), n = 1, 2, \ldots,$ the recursive linear averaging

can be expressed as

$$y(n) = \frac{n-1}{n} y(n-1) + \frac{1}{n} x(n-1) \tag{1}$$

For $n \geq N$ the "block-oriented" linear averaging has the form of

$$X(n-N) = \frac{1}{N} \sum_{k=1}^{N} x(n-k) \tag{2}$$

while the block-recursive average can be written as

$$y(n) = \frac{n-N}{n} y(n-N) + \frac{N}{n} X(n-N). \tag{3}$$

If (3) is evaluated only in every $N$th step, i.e. it is maximally decimated, then we can replace (3) with $n = mN, \quad m = 1, 2, \ldots,$ by

$$y(mN) = \frac{m-1}{m} y[(m-1)N] + \frac{1}{m} X[(m-1)N] \tag{4}$$

or simply

$$y(m) = \frac{m-1}{m} y(m-1) + \frac{1}{m} X(m-1) \tag{5}$$

where $m$ stands as block identifier. Note the formal correspondence with (1).

If the block identifier $m$ in equation (5) is replaced by a constant $Q > 1$ then an exponential averaging effect is achieved. This change makes the above block-oriented filter time-invariant and thus a frequency-domain characterization is also possible. In many practical applications exponential averaging provides the best compromise if both the noise reduction and the signal tracking capabilities are important. This is valid in our case as well, however, in this paper only the linear and the sliding averagers are investigated because they can be used directly to extend the size of certain signal transformation channels and can be applied in anytime systems.

A similar development can be provided for the sliding-window averagers. The recursive form of this algorithm is given for a block size of $N$ by

$$y(n) = y(n-1) + \frac{1}{N}[x(n-1) - x(n-N-1)]. \tag{6}$$

If in (6) the input samples are replaced by preprocessed data, e.g. as in (2), then a block-recursive form is also possible:

$$y(n) = y(n-N) + [X(n-N) - X(n-2N)] \tag{7}$$

which, however, has no practical meaning, since it gives back (2). But if the window size is integer multiple of $N$, e.g. $MN$, then the form

$$y(n) = y(n-N) + \frac{1}{M}[X(n-N) - X(n-(M+1)N) \tag{8}$$

has real importance. If (8) is evaluated only in every $N$th step, i.e. it is maximally decimated, then we can replace (8) with $n = mN$, $m = 1, 2, \ldots$, by

$$y(mN) = y[(m-1)N] + \frac{1}{M}[X((m-1)N) - X((m-M-1)N)] \qquad (9)$$

or simply

$$y(m) = y(m-1) + \frac{1}{M}[X(m-1) - X(m-M-1)] \qquad (10)$$

where $m$ stands as block identifier. Note the formal correspondence with (6).

The generalization of these averaging schemes to signal transformations and/or filter-banks is straightforward. Only (2) should be replaced by the corresponding "block-oriented" operation. *Fig. 1* shows the block diagram of the linear averaging scheme. This is valid also for the exponential averaging except $m$ must be replaced by $Q$. These frameworks can incorporate a variety of possible transformations and corresponding filter-banks which permit decimation by the block-size. Standard references, e.g. [6] provide the necessary theoretical and practical background.

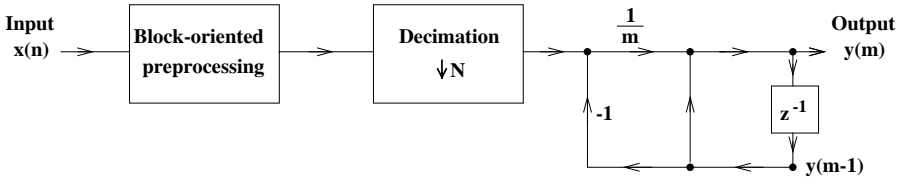


*Fig. 1.* Block-recursive linear averaging signal processing scheme, $n=mN$

The idea of transform-domain signal processing proved to be very efficient especially in adaptive filtering (see e.g. [7]). The contribution of this paper is directly applicable for the majority of these intensively cited algorithms. The most important practical advantage here compared to other methods is the early availability of rough estimates which can orientate in making decisions concerning further processing. The multiple-block sliding-window technique can be mentioned as a very characteristic algorithm of the proposed family. For this the computational complexity figures are also advantageous since using conventional methods to evaluate in "block-sliding-window" mode the transform of a block of $MN$ samples would require $M$ times an $(MN)*(MN)$ transformation, while the block-recursive solution calculates only for the last input block of $N$ samples, i.e. $M$ times an $(MN)*(N)$ "transformation".

As block-oriented preprocessing the DFT is the most widely used transformation for its fast algorithms (FFTs) and relatively easy interpretation. The above schemes can be operated for every "channel" of the DFT and after averaging this will correspond to the channel of a larger scale DFT. If linear averager is applied this scale equals $mN$ while for sliding averager this figure is $MN$. The number of

the channels obviously remains $N$ unless further parallel DFTs are applied. These additional DFTs have to locate their channel to the positions not covered by the existing channels. For the case where $M$=2, i.e. only one additional parallel DFT is needed, where this positioning can be solved with the so-called complementary DFT which is generated using the $N$th roots of -1. This DFT locates its channels into the positions $\pi/N$, $3\pi/N$, etc. For $M > 2$ proper frequency transposition techniques must be applied. If e.g. $M$=4 then the full DFT will be of size $4N$ and four $N$-point DFTs (working on complex data) are to be used. The first DFT is responsible for the channels in positions 0, $8\pi/4N$, etc. The second DFT should cover the $2\pi/4N$, $10\pi/4N$, etc., the third the $4\pi/4N$, $12\pi/4N$, etc, and finally the fourth the $6\pi/4N$, $14\pi/4N$, etc. positions, respectively. The first DFT does not need extra frequency transposition. The second and the fourth process complex input data coming from a complex modulator which multiplies the input samples by $e^{j2\pi n/4N}$ and $e^{j6\pi n/4N}$, respectively. The third DFT should be a complementary DFT.

It is obvious from the above development that if a full DFT is required the sliding-window DFT must be preferred otherwise the number of the parallel channels should grow by $m$.

The majority of the transform-domain signal processing methods prefers the DFT to other possible transformations. However, there are certain applications where other orthogonal transformations can also be utilized possibly with much better overall performance. Just to mention one example, consider the case when besides the transformed domain representation compression and transmission is also aimed.

Compact representations are useful for compression. If all the basis vectors have the same norm, a good approximation of the signal can be generated using those components that have significant weights. The negligible components can be thresholded, i.e., set to zero, without substantially degrading the signal reconstruction. (An additional advantage of compression is the denoising effect since white noise is incompressible.) The optimization of the representation from transmission point of view is that the signal is represented by the minimum number of nonzero coefficients (minimum load on the channel) at a given accuracy. Considering the characteristics of the signal to be represented other orthogonal transformations, like Walsh-Hadamard transformation can result in better performance. This feature can be of even more importance when nonstationary signals are to be represented (see e.g. [8]).

A further aspect of practical interest can be the end-to-end delay of the block-oriented processing. The time-recursive transformation algorithms described e.g. in [9] and [10] are sliding-window transformations, i.e. filter-banks providing transform domain representation of the last input data block in every step. Decimation is not "inherent" as it is the case if the transformation is considered as a serial to parallel conversion, therefore the processing rate can be either the input rate, the maximally decimated one, or any other in between. These techniques are not fast algorithms, however, "produce" less delay as those block-oriented algorithms, which start working only after the arrival of the complete input data block.

In [11] a fast polyphase filter-bank family has been reported which if maximally decimated has the same computational complexity as the fast transforms and additionally provides a uniform processing load along time. This approach seems to be advantageous if the end-to-end delay is to be minimized. The applicability of this approach to certain dedicated measurement problems has also been investigated [12].

## 4. Anytime Fourier Transformation

The above detailed algorithms can advantageously be applied in anytime systems. If the block-recursive linear averager ($L=mN$) (in case of sliding-window averager $MN$) is composed of $mN$-point DFTs then after the arrival of the first $N$ samples we will have a rough approximation of the signal, after $2N$ samples a better one, etc. The accuracy of the pre-results will not be exact, however the error is in most cases tolerable or even negligible.

In the followings two simple examples are presented illustrating the usability of the results. In the first example a 256-channel DFT is calculated recursively with $N$=64 for $m$=1,2,3,4. The input sequence applied was

$$x(n) = \cos\left(\frac{\pi(N+0.5)n}{2N}\right). \tag{11}$$

This single sinusoid is just in the middle between two measuring channels. The MATLAB simulations after processing the first, second, etc. blocks are given in *Fig. 2*.

In the second example a 256-channel DFT is calculated recursively with $N$=64 for $m$=1,2,8,16. The input sequence was

$$x(n) = \cos\left(\frac{\pi n}{2}\right) + rand - 0.5 \tag{12}$$

where *rand* stands for a random number generated by MATLAB between 0 and 1. The sinusoid is located exactly to a DFT channel position. The simulation results for $m$=1,2,8 and 16 are given on *Fig. 3*. The improvement in resolution and noise reduction is remarkable.

## 5. Accuracy Problems

Concerning anytime algorithms it is an important issue how can we handle accuracy problems. Accuracy can be worse both due to the lack of the appropriate input information and to the less accurate calculations. The characterization of the signal processing results is relatively easy if only the stationary responses are to be considered. Unfortunately, however, in time critical applications handling uncertainty issues in real-time can be of great importance. This means that the

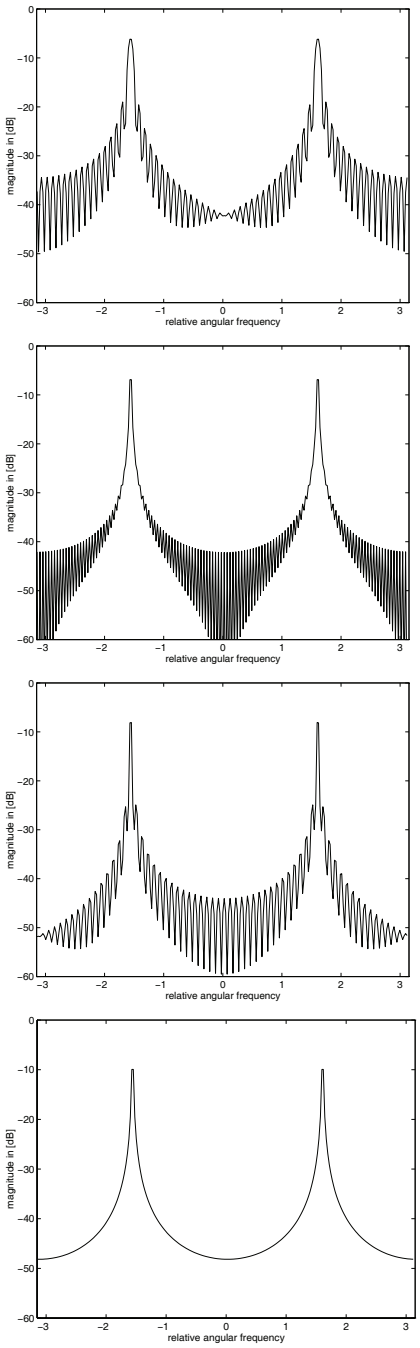


*Fig. 2.* 256-channel DFT of a single sinusoid with $N$=64, $m$=1,2,3,4, respectively

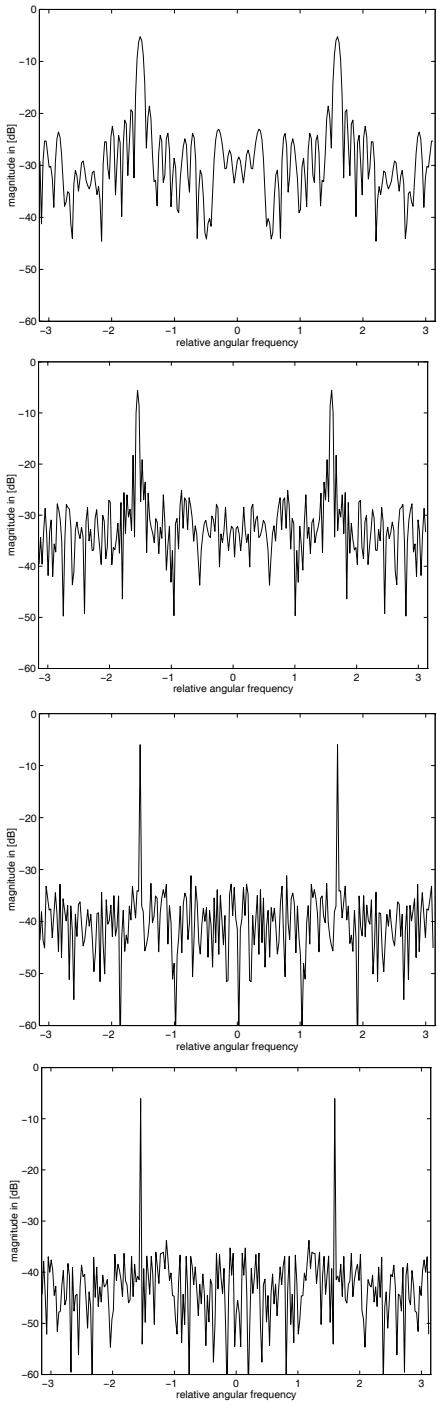


*Fig. 3.* 256-channel DFT of a single sinusoid plus noise. $N$=64, $m$=1,2,8,16, respectively

concept of anytime algorithms must be extended toward estimating the reliability of the calculated results. Such an extension should be able to provide error bounds, and similar uncertainty measures. In principle these measures can be generated using classical approaches, however, the elaboration of their conceptual background needs further efforts. How to characterize, e.g. a simple recursive averaging if some of the input samples fail to arrive in time? How to characterize the usual prediction-correction schemes from accuracy point of view if the correction can not be performed? How to continue if due to the lack of input the estimated outputs becomes useless? This topic is strongly related to some aspects of intelligent computing [13], where performance evaluation involves different accuracy issues, as well.

## 6. Conclusions

In this paper the concept of anytime Fourier analysis has been introduced. Anytime algorithms are a special class of adaptive algorithms. These adaptive algorithms are considered to improve the performance of signal processing under conditions of extreme computational burden. The digital signal evaluation algorithms of this class can tolerate temporal shortage of computational power and the lack of input data. The presented anytime Fourier analysis method is based on the combination of block-oriented and therefore typically fast algorithms with simple recursive averagers. It may improve on one hand the accuracy and/or resolution while on the other with the early availability of some rough estimates may reduce the side-effects of the delay caused by the block-oriented approach itself. The reduction of the delay is of real importance in applications where the response time is also specified. The idea of block-recursive filters and filter-banks can be extended toward higher-order averagers and other filtering effects, as well.

## Acknowledgement

This work was sponsored by the Hungarian Fund for Scientific Research (OTKA T 035190) and the Hungarian-Portugese Intergovern. S&T Cooperation Programme (P-24/03).

## References

[1] VÁRKONYI-KÓCZY, A. R. – KOVÁCSHÁZY, T., "Anytime Algorithms in Embedded Signal Processing Systems," *IX. European Signal Processing Conference, EUSIPCO-98*, Rhodes, Greece, Sep. 8-11, 1998, Vol. 1, pp. 169–172.

[2] BARON, C. – GEFFROY J.-C. – MOTET, G. EDS., *Embedded System Applications*, Kluwer Acad. Publ., 1997.

[3] ZILBERSTEIN, S., "Using Anytime Algorithms in Intelligent Systems," *AI Magazine*, **17**( 3), 1996, pp. 73-83.

[4] VÁRKONYI-KÓCZY, A. R. – KOVÁCSHÁZY, T. – TAKÁCS, O. – BENEDECSIK, Cs., "Anytime Algorithms in Intelligent Measurement and Control," *2000 World Automation Congress*, WAC'2000, Maui, USA, June 11-16, 2000, pp. ISIAC-156.1-6.

[5] VÁRKONYI-KÓCZY, A. R. – RUANO, A. – BARANYI, P. – TAKÁCS, O., "Anytime Information Processing Based on Fuzzy and Neural Network Models," *2001 IEEE Instrumentation and Measurement Technology Conference, IMTC/2001*, Budapest, Hungary, May 21-23, 2001, pp. 1247–1252.

[6] CROCHIERE, R. E. –RABINER, L.R., *Multirate Digital Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, N.J, 1983.

[7] SHYNK, J. J., "Frequency-Domain and Multirate Adaptive Filtering," *IEEE Signal Processing Magazine*, Jan. 1992, pp. 15–37.

[8] VÁRKONYI-KÓCZY, A. R. – FÉK, M. "Recursive Overcomplete Signal Representations," *IEEE Trans. on Instrumentation and Measurement*, 50(6), Dec. 2001, pp. 1698-1703.

[9] PÉCELI, G., "A Common Structure for Recursive Discrete Transforms," *IEEE Trans. on Circuits and Systems*, **33** (Oct. 1986), pp. 1035–1036.

[10] PADMANABHAN, M. – MARTIN, K. – PÉCELI, G. *Feedback-Based Orthogonal Filters*, Kluwer Academic Publishers, Boston/London/Dordrecht, 1996.

[11] VÁRKONYI-KÓCZY, A. R., "A Recursive Fast Fourier Transformation Algorithm," *IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing*, **42**(9), Sept. 1995, pp. 614-616.

[12] VÁRKONYI-KÓCZY, A. R., "Multi-Sine Synthesis and Analysis Via Walsh-Hadamard Transformation," *1996 IEEE International Symposium on Circuits and Systems, ISCAS'96*, May 12-15, Atlanta, Vol. 2, pp. 457–460.

[13] VÁRKONYI-KÓCZY, A .R. – PÉCELI, G.– DOBROWIECKI, T.P.– TAKÁCS, O. "Measurement Technology of Intelligent Systems," *1998 IEEE International Conference on Intelligent Engineering Systems, INES'98*, Vienna, Austria, Sep. 17-19, 1998, pp. 281–284.